PromptBERT: Improving BERT Sentence Embeddings with Prompts

Anonymous ACL submission

Abstract

The poor performance of the original BERT[1] for sentence semantic similarity has been widely discussed in previous works. We find that unsatisfactory performance is mainly due to the static token embeddings biases and the ineffective BERT layers, rather than the high cosine similarity of the sentence embeddings. To this end, we propose a prompt based sentence embeddings method which can reduce token embeddings biases and make the original BERT layers more effectively. By reformulating the sentence embeddings task as the fillin-the-blanks problem, our method significantly improves the performance of original BERT. We discuss two prompt representing methods and three prompt searching methods for prompt based sentence embeddings. Moreover, we propose a novel unsupervised training objective by the technology of template denoising, which substantially shortens the performance gap between the supervised and unsupervised setting. For experiments, we evaluate our method on both non fine-tuned and fine-tuned settings. Even a non fine-tuned method can outperform the fine-tuned methods like unsupervised ConSERT on STS tasks. Our fine-tuned method outperforms the state-of-the-art method SimCSE in both unsupervised and supervised settings. Compared to SimCSE, we achieve 2.29 and 2.58 points improvements on BERT and RoBERTa respectively under the unsupervised setting.

1 Introduction

In recent years, we have witnessed the success of pre-trained language models like BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) in sentence embeddings (Gao et al., 2021; Yan et al., 2021). However, the original BERT still shows poor performance in sentence embeddings (Reimers and Gurevych, 2019; Li et al., 2020). The most commonly used example is that it underperforms the traditional word embedding methods like GloVe (Pennington et al., 2014).

Previous research has linked anisotropy to explain the poor performance of the original BERT (Li et al., 2020; Yan et al., 2021; Gao et al., 2021). Anisotropy makes the token embeddings occupy a narrow cone, resulting in a high similarity between any sentence pair (Li et al., 2020). Li et al. (2020) proposed a normalizing flows method to transform the sentence embeddings distribution to a smooth and isotropic Gaussian distribution and Yan et al. (2021) presented a contrastive framework to transfer sentence representation. The goal of these methods is to eliminate anisotropy in sentence embeddings. However, we find that anisotropy is not the primary cause of poor semantic similarity. For example, averaging the last layer of the original BERT is even worse than averaging its static token embeddings in semantic textual similarity task, but the sentence embeddings from last layer are less anisotropic than static token embeddings.

Following this result, we find the original BERT layers[2] actually damage the quality of sentence embeddings. However, if we treat static token embeddings[3] as word embedding, it still yields unsatisfactory results compared to GloVe. Inspired by (Li et al., 2020), who found token frequency biases its distribution, we find the distribution is not only biased by frequency, but also case sensitive and subword in WordPiece (Wu et al., 2016). We design a simple experiment to test our conjecture by simply removing these biased tokens (e.g., high frequency subwords and punctuation) and using the average of the remaining token embeddings as sentence representation. It can outperform the

[1] In this paper, we use "original BERT" to refer to the BERT like models, which are not fine-tuned on downstream tasks.

[2] We denote the transformer blocks in BERT as BERT layers.

[3] We denote the BERT token embeddings as static token embeddings.

Glove and even achieve results comparable to post-processing methods BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021).

Motivated by these findings, avoiding embedding bias can improve the performance of sentence representations. However, it is labor-intensive to manually remove embedding biases and it may result in the omission of some meaningful words if the sentence is too short. Inspired by (Brown et al., 2020), which has reformulated the different NLP tasks as fill-in-the-blanks problems by different prompt, we propose a prompt based method by using the template to obtain the sentence representations in BERT. Prompt based method can avoid embedding bias and utilize the original BERT layers. We find original BERT can achieve reasonable performance with the help of the template in sentence embeddings, and it even outperforms some BERT based methods, which fine-tune BERT in down-stream tasks.

Our approach is equally applicable to fine-tuned setting. Current methods utilize the contrastive learning to help the BERT learn better sentence embeddings (Gao et al., 2021; Yan et al., 2021). However, the unsupervised methods still suffer from leaking proper positive pairs. Yan et al. (2021) discuss four data augmentation methods, but the performance seems worse than directly using the dropout in BERT as noise (Gao et al., 2021). We find the prompts can provide a better way to generate positive pairs by different viewpoints from different templates. To this end, we propose a prompt based contrastive learning method with template denoising to leverage the power of BERT in an unsupervised setting, which significantly shortens the gap between the supervised and unsupervised performance. Our method achieves the state-of-the-art results in both unsupervised and supervised settings.

2 Related Work

Learning sentence embeddings as a fundamental NLP problem has been largely studied. Currently, how to leverage the power of BERT in sentence embeddings has become a new trend. Many works (Li et al., 2020; Gao et al., 2021) achieved strong performance with BERT in both supervised and unsupervised settings. Among these works, contrastive learning based methods achieve the state-of-the-art results. These works (Gao et al., 2021; Yan et al., 2021) pay attention to constructing positive sentence pairs. Gao et al. (2021) proposed a novel contrastive training objective to directly use inner dropout as noise to construct positive pairs. Yan et al. (2021) discuss four methods to construct positive pairs.

Although BERT achieved great success in sentence embeddings, the original BERT shows unsatisfactory performance. Contextual token embeddings from original BERT even underperform the word embeddings like GloVe. One explanation is the anisotropy in the original BERT, which causes sentence pairs to have high similarity. Following this explanation, BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021) have been proposed to reduce the anisotropy by post-processing the sentence embeddings from original BERT.

3 Rethinking the Sentence Embeddings of Original BERT

Previous works (Yan et al., 2021; Gao et al., 2021) explained the poor performance of original BERT is limited by the learned anisotropic token embeddings space, where the token embeddings occupy a narrow cone. However, we find that anisotropy is not a key factor to inducing poor semantic similarity by examining the relationship between the aniostropy and performance. We think the main reasons are the ineffective BERT layers and static token embedding biases.

Observation 1: Original BERT layers fail to improve the performance. In this section, we analyze the influence of BERT layers by comparing the two sentence embeddings methods: averaging static token embeddings (input of the BERT layers) and averaging last layer (output of the BERT layers). We report the sentence embeddings performance and its sentence level anisotropy.

To measure the anisotropy, we follow the work of (Ethayarajh, 2019) to measure the sentence level anisotropy in sentence embeddings. Let s_i be a sentence that appears in corpus $\{s_1, ..., s_n\}$. The anisotropy can be measured as follows:

$$\frac{1}{n^2 - n} \left| \sum_i \sum_{j \neq i} \cos\left(M(s_i), M(s_j)\right) \right| \quad (1)$$

where M denotes the sentence embeddings method, which maps the raw sentence to its embedding and cos is the cosine similarity. In other words, the anisotropy of M is measured by the average cosine similarity of a set of sentences. If sen-

tence embeddings was isotropic (i.e., directionally uniform), then the average cosine similarity between uniformly randomly sampled sentences would be 0 (Arora et al., 2016). The closer it is to 1, the more anisotropic the embedding of sentences. We randomly sample 100,000 sentences from the Wikipedia corpus to compute the anisotropy.

We compare different pre-trained models (*bert-base-uncased*, *bert-base-cased* and *roberta-base*) and different sentence embeddings methods (last layer avgerage, averaging of last hidden layer tokens as sentence embeddings and static token embeddings, directly averaging of static token embeddings). We have shown the spearman correlation, sentence level anisotropy of these methods in Table 1.

Pre-trained models	Correlation	Sentence anisotropy
Static token embeddings avg.		
bert-base-uncased	56.02	0.8250
bert-base-cased	56.65	0.5755
roberta-base	55.88	0.5693
Last layer avg.		
bert-base-uncased	52.57	0.4874
bert-base-cased	56.93	0.7514
roberta-base	53.49	0.9554

Table 1: The spearman correlation, sentence anisotropy of Last layer average. and Static token embeddings average. The spearman correlation is the average of correlation on STS12-16, STS-B and SICK.

As Table 1 shows, we find the BERT layers in *bert-base-uncased* and *roberta-base* significantly harm the sentence embeddings performance. Even in *bert-base-cased*, the gain of BERT layers is trivial with only 0.28 improvement. We also show the sentence level anisotropy of each method. The performance degradation of the BERT layers seems not to be related to the sentence level anisotropy. For example, the last layer average is more isotropic than the static token embeddings average in *bert-base-uncased*. However, the static token embeddings average achieves better sentence embeddings performance.

Observation 2: Embedding biases harms the sentence embeddings performance. Li et al. (2020) found that token embeddings can be biased to token frequency. Similar problems have been studied in (Yan et al., 2021). The anisotropy in BERT static token embeddings is sensitive to token frequency. Therefore, we investigate whether embedding bias yields unsatisfactory performance of sentence embeddings. We observe that the token embeddings is not only biased by token frequency, but also subwords in WordPiece (Wu et al., 2016) and case sensitive.

As shown in Figure 1, we visualize these biases in the token embeddings of *bert-base-uncased*, *bert-base-cased* and *roberta-base*. The token embeddings of three pre-trained models are highly biased by the token frequency, subword and case. The token embeddings can roughly divided into three regions according to the subword and case biases : 1) the lowercase begin-of-word tokens, 2) the uppercase begin-of-word tokens and 3) the subword tokens. For uncased pre-trained model *bert-base-uncased*, the token embeddings also can roughly divided into two regions: 1) the begin-of-word tokens, 2) the subword tokens.

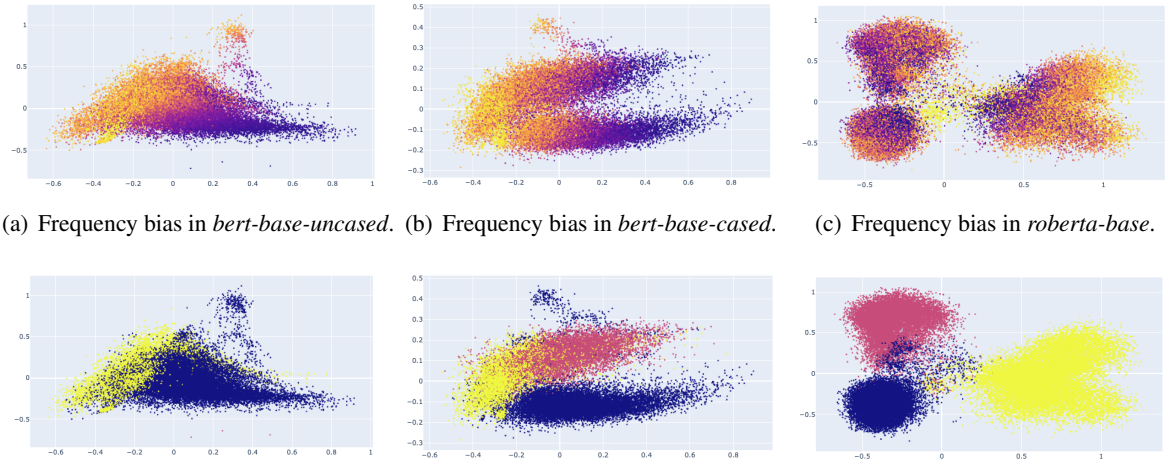
For frequency bias, we can observe that high frequency tokens are clustered, while low frequency tokens are dispersed sparsely in all models (Yan et al., 2021). The begin-of-word tokens are more vulnerable to frequency than subword tokens in BERT. However, the subword tokens are more vulnerable in RoBERTa.

Previous works (Yan et al., 2021; Li et al., 2020) often connect the concept of "token embeddings bias" with the token embeddings anisotropy as the reason for bias. However, we think the anisotropy is unrelated to the bias. The bias means the distribution of embedding is disturbed by some irrelevant information like token frequency, which can be directly visualized according to the PCA. For the anisotropy, it means the whole embedding occupies a narrow cone in the high dimensional vector space, which cannot be directly visualized.

M	average cosine similarity
bert-base-uncased	0.4445
bert-base-cased	0.1465
roberta-base	0.0235

Table 2: The average cosine similarity in static token embeddings

Table 2 shows the static token embeddings anisotropy of three pre-trained models in Figure 1 according to the average the cosine similarity between any two token embeddings. Contrary to the previous conclusion (Yan et al., 2021; Li et al., 2020), we find only *bert-base-uncased*'s static token embeddings is highly anisotropic. The static



(a) Frequency bias in *bert-base-uncased*. (b) Frequency bias in *bert-base-cased*. (c) Frequency bias in *roberta-base*.

(d) Subword and Case biases in *bert-base-uncased*. (e) Subword and Case biases in *bert-base-cased*. (f) Subword and Case biases in *roberta-base*.

Figure 1: 2D visualization of token embeddings with different biases. For frequency bias, the darker the color, the higher the token frequency. For subword and case bias, yellow represents subword and red represents the token contains capital letters.

token embeddings like *roberta-base* are isotropic with 0.0235 average cosine similarity. For biases, these models are suffered from the biases in static token embeddings, which is irrelevant to the anisotropy.

	cased	uncased	roberta
Static Token Embeddings	56.93	56.02	55.88
– Freq.	60.27	59.65	65.41
– Freq. & Sub.	64.83	62.20	64.89
– Freq. & Sub. & Case	65.07	-	65.06
– Freq. & Sub. & Case & Pun.	66.05	63.10	67.64

Table 3: The influence of static embedding biases in spearman correlation. The spearman correlation is the average of STS12-16, STS-B and SICK. Cased, uncased and roberta represent *bert-base-cased*, *bert-base-uncased* and *roberta-base*. For Freq., Sub., Case. and Pun., we remove the top frequency tokens, subword tokens, uppercase tokens and punctuation respectively. More details can be found in Appendix A.

To prove the negative impact of biases, we show the influence of biases to the sentence embeddings with averaging static token embeddings as sentence embeddings (without BERT layers). The results of eliminating embedding biases are quite impressive on three pre-trained models in Table 3. Simply removing a set of tokens, the result can be improved by 9.22, 7.08 and 11.76 respectively. The final result of *roberta-base* can outperform post-processing methods such as BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021) with only using static token embeddings.

Manually removing embedding biases is a simple method to improve the performance of sentence embeddings. However, if the sentence is too short, this is not an adequate solution, which may result in the omission of some meaningful words.

4 Prompt Based Sentence Embeddings

Inspired by (Brown et al., 2020), we propose a prompt based sentence method to obtain sentence embeddings. By reformulating the sentence embedding task as the mask language task, we can effectively use the original BERT layers by leveraging the large-scale knowledge. We also avoid the embedding biases by representing sentences from [MASK] tokens.

However, unlike the text classification or question-answering tasks, the output in sentence embeddings is not the label tokens predicted by MLM classification head, but the vector to represent the sentence. We follow these two problems to discuss the implementation of prompt based sentence embeddings: 1) how to represent sentences with the prompt, and 2) how to find a proper prompt for sentence embeddings. Based on these, we propose a prompt based contrastive learning method to fine-tuning BERT on sentence embeddings.

4.1 Represent Sentence with the Prompt

In this section, we discuss two methods to represent one sentence with a prompt. For example, we have a template "[X] means [MASK]", where [X] is a

placeholder to put sentences and [MASK] represent the [MASK] token. Given a sentence x_{in}, we map x_{in} to x_{prompt} with the template. Then we feed x_{prompt} to a pre-trained model to generate sentence representation $\mathbf{h}$.

One method is to use the hidden vector of [MASK] token as sentence representation:

$$\mathbf{h} = \mathbf{h}_{[\text{MASK}]} \tag{2}$$

For the second method like other prompt based tasks, we get the top-k tokens according to $\mathbf{h}_{[\text{MASK}]}$ and MLM classification head, then find the weighted average of these tokens according to probability distribution. The $\mathbf{h}$ can be formulated as:

$$\mathbf{h} = \frac{\sum_{v \in \mathcal{V}_{\text{top-}k}} \mathbf{W}_v \, \mathrm{P}\left([\text{MASK}] = v | \mathbf{h}_{[\text{MASK}]}\right)}{\sum_{v \in \mathcal{V}_{\text{top-}k}} \mathrm{P}\left([\text{MASK}] = v | \mathbf{h}_{[\text{MASK}]}\right)} \tag{3}$$

where v is the BERT token in the top-k tokens set $\mathcal{V}_{\text{top-}k}$, $\mathbf{W}_v$ is the static token embeddings of v and $\mathrm{P}\left([\text{MASK}] = v | \mathbf{h}_{[\text{MASK}]}\right)$ denotes the probability of token v be predicted by MLM head with $\mathbf{h}_{[\text{MASK}]}$.

The second method, which maps the sentence to the tokens, is more conventional than the first. But its disadvantages are obvious: 1) as previously noted, due to the sentence embeddings from averaging of static token embeddings, it still suffers from biases. 2) weight averaging makes the BERT hard to fine-tune in down-stream tasks. For these reasons, we represent the sentence with the prompt by the first method.

4.2 Prompt Search

For prompt based tasks, one key challenge is to find templates. We discuss three methods to search template in this section: manual search, template generation based T5 (Gao et al., 2020) and OptiPrompt (Zhong et al., 2021). We use the spearman correlation in the STS-B development set as the main metric to evaluate different templates.

For manual search, we need to hand-craft templates, which encourage the whole sentence to be represented in $\mathbf{h}_{[\text{MASK}]}$. To search templates, we divide the template into two parts: relationship tokens, which denotes the relationship between [X] and [MASK], and prefix tokens, which wraps [X]. Then we greedily search for templates following the relationship tokens and prefix tokens.

Some results of greedy searching are shown in Table 4. When it comes to sentence embeddings,

Template	STS-B dev.
Searching for relationship tokens	
[X] [MASK] .	39.34
[X] is [MASK] .	47.26
[X] mean [MASK] .	53.94
[X] means [MASK] .	63.56
Searching for prefix tokens	
This [X] means [MASK] .	64.19
This sentence of [X] means [MASK] .	68.97
This sentence of "[X]" means [MASK] .	70.19
This sentence : "[X]" means [MASK] .	73.44

Table 4: Greedy searching templates on *bert-base-uncased*.

different templates produce extremely varied results. Compared to simply concatenating the [X] and [MASK], complex templates like *This sentence : "[X]" means [MASK].*, can improve the spearman correlation by 34.10.

For template generation based on T5, Gao et al. (2020) proposed a novel method to automatically generate templates by using T5 to generate templates according to the sentences and corresponding labels. The generated templates can outperform the manual searched templates in the GLUE benchmark (Wang et al., 2018).

However, the main issue to implement it is the lack of label tokens. Tsukagoshi et al. (2021) successfully transformed the sentence embeddings task to the text classification task by classifying the definition sentence to its word according to the dictionary. Inspired by this, we use words and corresponding definitions to generate 500 templates (e.g., orange: a large round juicy citrus fruit with a tough bright reddish-yellow rind). Then we evaluate these templates in the STS-B development set, the best spearman correlation is 64.75 with the template "Also called [MASK]. [X]". Perhaps it is the gap between sentence embeddings and word definition. This method cannot generate better templates compared to manual searching.

OptiPrompt (Zhong et al., 2021) replaced discrete template with the continuous template. To optimize the continuous template, we use the unsupervised contrastive learning as training objective following the settings in (Gao et al., 2021) with freezing the whole BERT parameters, and the continuous template is initialized by manual template's static token embeddings. Compared to the input manual template, the continuous template can increase the spearman correlation from 73.44 to 80.90 on STS-B development set.

4.3 Prompt Based Contrastive Learning with Template Denoising

Recently, contrastive learning successfully leverages the power of BERT in sentence embeddings. A challenge in sentence embeddings contrastive learning is how to construct proper positive instances. Gao et al. (2021) directly used the dropout in the BERT as positive instances. Yan et al. (2021) discussed the four data augmentation strategies such as adversarial attack, token shuffling, cutoff and dropout in the input token embeddings to construct positive instances. Motivated by the prompt based sentence embeddings, we propose a novel method to reasonably generate positive instances based on prompt.

The idea is using the different templates to represent the same sentence as different points of view, which helps model to produce more reasonable positive pairs. In order to reduce the influence of the template itself on the sentence representation, we propose a novel way to denoise the template information. Given the sentence x_i, we first calculate the corresponding sentence embeddings $\mathbf{h}_i$ with a template. Then we calculate the template bias $\hat{\mathbf{h}}_i$ by directly feeding BERT with the template and the same template position ids. For example, if the x_i has 5 tokens, then the position ids of template tokens after the [X] will be added by 5 to make sure the position ids of template is same. Finally, we can directly use the $\mathbf{h}_i - \hat{\mathbf{h}}_i$ as the denoised sentence representation. For the template denoising, more details can be found in Discussion.

Formally, let $\mathbf{h}_i^{'}$ and $\mathbf{h}_i$ denote the sentence embeddings of x_i with different templates, $\hat{\mathbf{h}}_i^{'}$ and $\hat{\mathbf{h}}_i$ denotes the two template biases of the x_i respectively, the final training objective is as follows:

$$\ell_i = -\log \frac{e^{\cos\left(\mathbf{h}_i - \hat{\mathbf{h}}_i, \mathbf{h}_i^{'} - \hat{\mathbf{h}}_i^{'}\right)/\tau}}{\sum_{j=1}^{N} e^{\cos\left(\mathbf{h}_i - \hat{\mathbf{h}}_i, \mathbf{h}_j^{'} - \hat{\mathbf{h}}_j^{'}\right)/\tau}} \quad (4)$$

where τ is a temperature hyperparameter in contrastive learning and N is the size of mini-batch.

5 Experiments

We conduct experiments on STS tasks with non fine-tuned and fine-tuned BERT settings. For non fine-tuned BERT settings, we exploit the performance of original BERT in sentence embeddings, which corresponds to the previous findings of the poor performance of original BERT. For fine-tuned BERT settings, we report the unsupervised and supervised results by fine-tuning BERT with downstream tasks. The results of transfer tasks are in Appendix B.

5.1 Dataset

Following the past works (Yan et al., 2021; Gao et al., 2021; Reimers and Gurevych, 2019), we conduct our experiments on 7 common STS datasets: STS tasks 2012-2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016) STS-B(Cer et al., 2017), SICK-R (Marelli et al., 2014). We use the SentEval toolkit (Conneau and Kiela, 2018) to download all 7 datasets. The sentence pairs in each datasets are scored from 0 to 5 to indicate semantic similarity.

5.2 Baselines

We compare our method with both enlightening and state-of-the-art methods. To validate the effectiveness of our method in the non fine-tuned setting, we use the GLoVe (Pennington et al., 2014) and post-process methods: BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021) as baselines. For the fine-tuned setting, we compare our method with IS-BERT(Zhang et al., 2020), InferSent(Conneau et al., 2017), Universal Sentence Encoder(Cer et al., 2018), SBERT(Reimers and Gurevych, 2019) and the contrastive learning based methods: SimCSE (Gao et al., 2021) and ConSERT (Yan et al., 2021).

5.3 Implementation Details

For the non fine-tuned setting, we report the result of BERT to validate the effectiveness of our representation method. For the fine-tuned setting, we use BERT and RoBERTa with the same unsupervised and supervised training data with (Gao et al., 2021). Our methods are trained with prompt based contrastive learning with template denosing. The templates used for both settings are manual searched according to Table 4. More details can be found in Appendix C.

5.4 Non Fine-Tuned BERT Results

To connect with the previous analysis of the poor performance of original BERT, we report our prompt based methods with non fine-tuned BERT in Table 5. Using templates can substantially improve the results of original BERT on all datasets. Compared to pooling methods like averaging of last layer or averaging of first and last layers, our methods can improve spearman correlation by more

Method	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
GLoVe embeddings avg.†	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
BERT last avg.	30.87	59.89	47.73	60.29	63.73	47.29	58.22	52.57
static avg.	42.38	56.74	50.60	65.08	62.39	56.82	58.15	56.02
first-last avg.†	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
static remove biases avg.	53.09	66.48	65.09	69.80	67.85	61.60	57.80	63.10
BERT-flow †	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
BERT-whitening †	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
Prompt based BERT (manual)	60.96	73.83	62.18	71.54	68.68	70.60	67.16	67.85
Prompt based BERT (manual&OptiPrompt)	**64.56**	**79.96**	**70.05**	**79.37**	**75.35**	**77.25**	**68.56**	**73.59**

Table 5: The performance comparison of our unfine-tuned BERT method on STS tasks. †: results from (Gao et al., 2021). The BERT-flow(Li et al., 2020) and BERT-whitening (Su et al., 2021) use the "NLI" setting. All BERT based methods use *bert-base-uncased* . Last avg. denotes averaging the last layer of BERT. Static avg. denotes averaging the static token embedding of BERT. First-last avg. (Su et al., 2021) uses the first and last layer. Static remove biases avg. means removing biased tokens in static avg., which we have introduced before.

Method	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Unsupervised models								
IS-BERT$_{\text{base}}$ ¶	56.77	69.24	61.21	75.23	70.16	69.21	64.25	66.58
ConSERT$_{\text{base}}$ ‡	64.64	78.49	69.07	79.72	75.95	73.97	67.31	72.74
SimCSE-BERT$_{\text{base}}$ ‡	68.40	82.41	74.38	80.91	78.56	76.85	**72.23**	76.25
PromptBERT$_{\text{base}}$	**71.56**$_{\pm 0.18}$	**84.58**$_{\pm 0.22}$	**76.98**$_{\pm 0.26}$	**84.47**$_{\pm 0.24}$	**80.60**$_{\pm 0.21}$	**81.60**$_{\pm 0.22}$	69.87$_{\pm 0.40}$	**78.54**$_{\pm 0.15}$
RoBERTa$_{\text{base}}$-whitening†	46.99	63.24	57.23	71.36	68.99	61.36	62.91	61.73
SimCSE-RoBERTa$_{\text{base}}$ †	70.16	81.77	73.24	81.36	80.65	80.22	68.56	76.57
PromptRoBERTa$_{\text{base}}$	**73.94**$_{\pm 0.90}$	**84.74**$_{\pm 0.36}$	**77.28**$_{\pm 0.41}$	**84.99**$_{\pm 0.25}$	**81.74**$_{\pm 0.29}$	**81.88**$_{\pm 0.37}$	**69.50**$_{\pm 0.57}$	**79.15**$_{\pm 0.25}$
Supervised models								
InferSent-GloVe§	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
SBERT$_{\text{base}}$ §	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT$_{\text{base}}$-flow†	69.78	77.27	74.35	82.01	77.46	79.12	76.21	76.60
SBERT$_{\text{base}}$-whitening†	69.65	77.57	74.66	82.27	78.39	79.52	76.91	77.00
ConSERT$_{\text{base}}$ ‡	74.07	83.93	77.05	83.66	78.76	81.36	76.77	79.37
SimCSE-BERT$_{\text{base}}$ †	75.30	84.67	80.19	85.40	80.82	84.25	80.39	81.57
PromptBERT$_{\text{base}}$	**75.48**	**85.59**	**80.57**	**85.99**	**81.08**	**84.56**	**80.52**	**81.97**
SRoBERTa$_{\text{base}}$ §	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa$_{\text{base}}$-whitening†	70.46	77.07	74.46	81.64	76.43	79.49	76.65	76.60
SimCSE-RoBERTa$_{\text{base}}$ †	76.53	85.21	80.95	86.03	82.57	85.83	**80.50**	82.52
PromptRoBERTa$_{\text{base}}$	**76.75**	**85.93**	**82.28**	**86.69**	**82.80**	**86.14**	80.04	**82.95**

Table 6: The performance comparison of our fine-tuned BERT methods on STS tasks. For unsupervised models, we found the result of unsupervised constrastive learning is unstable, and we train our model with 10 random seeds. †: results from (Gao et al., 2021). ‡: results from (Yan et al., 2021). §: results from (Reimers and Gurevych, 2019). ¶: results from (Zhang et al., 2020).

than 10%. Compared to the postprocess methods: BERT-flow and BERT-whitening, only using the manual template surpasses can these methods. Moreover, we can use the continuous template by OptiPrompt to help original BERT achieve much better results, which even outperforms unsupervised ConSERT in Table 6.

5.5 Fine-Tuned BERT Results

The results of fine-tuned BERT are shown in Table 6. Following previous works (Reimers and Gurevych, 2019), we run unsupervised and supervised methods respectively. Although the current contrastive learning based methods (Gao et al., 2021; Yan et al., 2021) achieved significant improvement compared to the previous methods, our method still outperforms them. Prompt based contrastive learning objective significantly shortens the gap between the unsupervised and supervised methods. It also proves our method can leverage the knowledge of unlabeled data with different templates as positive pairs. Moreover, we report the unsupervised performance with 10 random seeds to achieve more accurate results. In Discussion, we also report the result of SimCSE with 10 random seeds. Compared to SimCSE, our method shows more stable results than it.

Sentence	Top-5 tokens	Top-5 tokens after template denoising
i am sad.	sad,sadness,happy,love,happiness	sad,sadness,crying,grief,tears
i am not happy.	happy,happiness,sad,love,nothing	sad,happy,unhappy,upset,angry
the man is playing the guitar.	guitar,song,music,guitarist,bass	guitar,guitarist,guitars,playing,guitarists
the man is playing the piano.	piano,music,no,yes,bass	piano,pianist,pianos,playing,guitar

Table 7: The top-5 tokens predicted by manual template with original BERT.

5.6 Effectiveness of Prompt Based Contrastive Learning with Template Denoising

We report the results of different unsupervised training objectives in prompt based BERT. We use the following training objectives: 1) the same template, which uses inner dropout noise as data augmentation (Gao et al., 2021) 2) the different templates as positive pairs 3) the different templates with template denoising (our default method). Moreover, we use the same template and setting to predict and only change the way to generate positive pairs in the training stage. All results are from 10 random runs. The result is shown in Table 8. We observe our method can achieve the best and most stable results among three training objectives.

	$\text{BERT}_{\text{base}}$	$\text{RoBERTa}_{\text{base}}$
same template (dropout)	$78.16_{\pm 0.17}$	$78.16_{\pm 0.44}$
different templates	$78.19_{\pm 0.29}$	$78.17_{\pm 0.44}$
different templates with denoising	$78.54_{\pm 0.15}$	$79.15_{\pm 0.25}$

Table 8: Comparison of different unsupervised training objectives.

6 Discussion

6.1 Template Denoising

We find the template denoising efficiently removes the bias from templates and improves the quality of top-k tokens predicted by MLM head in original BERT. As Table 7 shows, we predict some sentences' top-5 tokens in the [MASK] tokens. We find the template denoising removes the unrelated tokens like "nothing,no,yes" and helps the model predict more related tokens. To quantify this, we also represent the sentence from the Eq. 3 by using the weighted average of top-200 tokens as the sentence embeddings. The results are shown in Table 9. The template denoising significantly improves the quality of tokens predicted by MLM head. However, it can't improve the performance for our default represent method in the Eq. 2 ([MASK] token in Table 9). In this work, we only use the template denoising in our contrastive training objective, which helps us eliminate different template biases.

	no denoising	denoising
avg. Top-200 tokens	56.19	60.39
[MASK] token	67.85	67.43

Table 9: Influence of template denoising in sentence embeddings.

6.2 Stability in Unsupervised Contrastive Learning

To prove the unstable results in unsupervised contrastive learning in sentence embeddings, we also reproduce the result of unsupervised $\text{SimCSE-BERT}_{\text{base}}$ with 10 random seeds in Table 10. Our results are more stable than SimCSE. The difference between the best and worst results can be up to 3.14% in SimCSE. However, the gap in our method is only 0.53.

	Mean	Max	Min
$\text{SimCSE-BERT}_{\text{base}}$	$75.42_{\pm 0.86}$	76.64	73.50
$\text{PromptBERT}_{\text{base}}$	$78.54_{\pm 0.15}$	78.86	78.33

Table 10: Results in unsupervised contrastive learning.

7 Conclusion

In this paper, we analyzed the poor performance of original BERT for sentence embeddings. The main reason is not the anisotropy, but the static token embeddings biases and ineffectively using original BERT layers. Based on these findings, we proposed a prompt based sentence embedding method to avoid static token embeddings biases and leverage the pre-trained knowledge in the original BERT layers. Our method significantly improved the performance of the original BERT. We also proposed a new prompt based contrastive learning method to shorten the gap between the unsupervised and supervised methods. Both our unsupervised and supervised methods achieve the state-of-the-art performance.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from bert for semantic textual similarity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Hayato Tsukagoshi, Ryohei Sasano, and Koichi Takeda. 2021. Defsent: Sentence embeddings using definition sentences. *arXiv preprint arXiv:2105.04339*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. *arXiv preprint arXiv:2009.12061*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*.

A Static Token Embeddings Biases

A.1 Eliminating Biases by Removing Tokens

We reported the detailed implementation of eliminating static token embeddings biases by deleting tokens on *bert-base-uncased*, *bert-base-cased* and *roberta-base*. For Freq. tokens, we follow the settings in (Yan et al., 2021) and remove the top 36 frequent tokens. The removed Freq. tokens are shown in Table 11. For Sub. tokens, we directly remove all subword tokens (yellow tokens in Figure 2). For Case. tokens, only SICK(Marelli et al., 2014) has sentences with upper and lower case, and we lowercase these sentences to remove the uppercased tokens (red tokens in Figure 2) . For Pun., we remove the tokens, which contain only punctuations.

A.2 Eliminating Biases by Pre-training

According to (Gao et al., 2019), we find the most of biases in static token embeddings are gradient from the MLM classification head weight, which transform the last hidden vector of [MASK] to the probability of all tokens. The tying weight between the static token embeddings and MLM classification head causes static token embeddings to suffer from bias problems.

	Removed Top frequency Tokens
bert-base-uncased	. a the in , is to of and ' on
and	- s with for " at s woman are
	two that you dog said playing
bert-base-cased	an as was from : by white
roberta-base	Ġ. Ġa Ġthe Ġin a Ġ, Ġis Ġto Ġof Ġon Ġ' s . the Ġman - Ġwith Ġfor Ġwoman Ġare Ġ" Ġthat Ġit Ġdog Ġplaying Ġwas Ġas Ġfrom Ġ: Ġyou i Ġby

Table 11: Removed top 36 frequent tokens in *bert-base-cased*, *bert-base-uncased* and *roberta-base*.

We have pre-trained two BERT-like models with the MLM pre-training objective. The only difference between the two pre-trained models is tying and untying the weight between static token embeddings and MLM classification head. We have pre-trained these two models on 125k steps with 2k batch sizes.

As shown in Figure 2, we have shown the static token embeddings of the untying model, MLM head weight of untying model and static token embeddings (MLM head weight) of the tying model. The distribution of the tying model and the head weight of the untying model is same with *bert-base-cased* in Figure 1, which severely suffers from the embedding biases. However, the distribution of the token embeddings in the untying weights model is less influenced by these biased. We also report the average spearman correlation of three embedding on STS tasks in Table 12. Static token embeddings of the untying model achieves the best correlation among the three embedding.

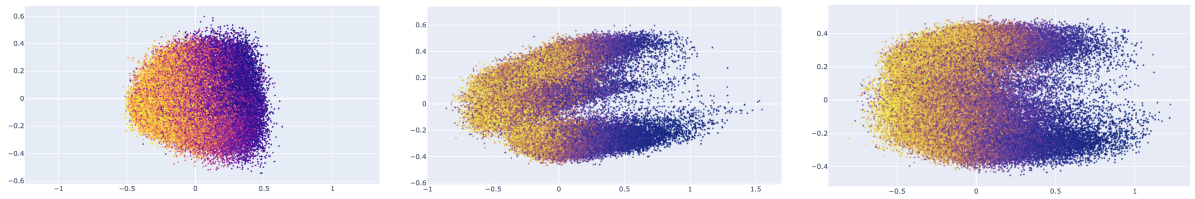
	Avg.
MLM head of untying model	43.33
Static token embeddings of untying model	49.41
Static token embeddings of tying model	45.68

Table 12: The avg. spearman correlation of three embeddings.

B Transfer Tasks

We also evaluate our models on the following transfer tasks: MR, CR, SUBJ, MPQA, SST-2, TREC and MRPC. We follow the default configurations in SentEval[4]. The results is shown in Table 13. Comparing to SimCSE, our RoBERTa based method

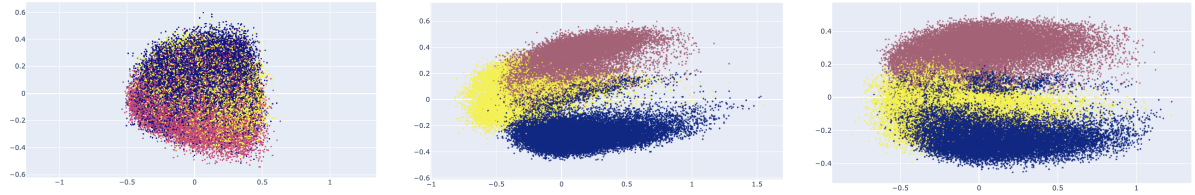
[4]https://github.com/facebookresearch/SentEval



(a) Frequency bias in static token embeddings of untying weights pre-trained model.

(b) Frequency bias in MLM head of untying weights pre-trained model.

(c) Frequency bias in tying weights pre-trained model.



(d) Subword and Case biases in static token embeddings of untying weights pre-trained model.

(e) Subword and Case biases in MLM head of untying weights pre-trained model.

(f) Subword and Case biases in tying weights pre-trained model.

Figure 2: 2D visualization of static token embeddings in untying and tying weights pre-trained model. For frequency bias, the darker the color, the higher the token frequency. For subword and case bias, yellow represents subword and red represents the token contains capital letters.

Method	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC	Avg.
Unsupervised models								
Avg. BERT embedding	78.66	86.25	94.37	88.66	84.40	92.80	69.54	84.94
BERT-[CLS] embedding	78.68	84.85	94.21	88.23	84.13	91.40	71.13	84.66
IS-BERT	81.09	**87.18**	**94.96**	88.75	**85.96**	88.64	74.24	85.83
SimCSE-BERT	**81.18**	86.46	94.45	88.88	85.50	**89.80**	74.43	**85.81**
PromptBERT	80.74	85.49	93.65	**89.32**	84.95	88.20	**76.06**	85.49
SimCSE-RoBERTa	81.04	87.74	**93.28**	86.94	86.60	84.60	73.68	84.84
PromptRoBERTa	**83.82**	**88.72**	93.19	**90.36**	**88.08**	**90.60**	**76.75**	**87.36**
Supervised models								
InferSent-GloVe	81.57	86.54	92.50	90.38	84.18	88.20	75.77	85.59
Universal Sentence Encoder	80.09	85.19	93.98	86.70	86.38	93.20	70.14	85.10
SBERT	**83.64**	**89.43**	94.39	89.86	**88.96**	**89.60**	76.00	**87.41**
SimCSE-BERT	82.69	89.25	**94.81**	89.59	87.31	88.40	73.51	86.51
PromptBERT	83.14	89.38	94.49	**89.93**	87.37	87.40	**76.58**	86.90
SRoBERTa	84.91	90.83	92.56	88.75	90.50	88.60	**78.14**	87.76
SimCSE-RoBERTa	84.92	**92.00**	94.11	89.82	91.27	88.80	75.65	88.08
PromptRoBERTa	**85.74**	91.47	**94.81**	**90.93**	**92.53**	**90.40**	77.10	**89.00**

Table 13: Transfer task results of different sentence embedding models.

can improve 2.52 and 0.92 on unsupervised and supervised models respectively.

C Training Details

For the non fine-tuned setting, the manual template we used is *This sentence : "[X]" means [MASK] .*. For OptPrompt, we first initialize the template embeddings with the manual template and then train these template embeddings by freezing BERT with the unsupervised training task followed by (Gao et al., 2021), and the batch size, learning-rate, epoch and valid steps are 256, 3e-5, 5 and 1000.

For the fine-tuned setting, all training data is same with (Gao et al., 2021). The max sentence sequence length is set to 32. For templates, we only use the manual templates, which are manually searched according to STS-B dev in unfine-tuned models. The templates is shown in Table 14. For unsupervised method, we use two different

templates for unsupervised training with template denosing according to our prompt based training objective. In predicting, we directly use the one template without template denoising . For supervised method, we use template denoising with same template for contrastive learning, because we already have supervised negative samples. We also report other training details in Table 15.

Model	Template
BERT	This sentence of "[X]" means [MASK] .
	This sentence : "[X]" means [MASK] .
RoBERTa	This sentence : '[X]' means [MASK] .
	The sentence : '[X]' means [MASK] .

Table 14: Templates for our method in fine-tuned setting

	Unsupervised		*Supervised*	
	BERT	RoBERTa	BERT	RoBERTa
Batch size	256	256	512	512
Learning rate	1e-5	1e-5	5e-5	5e-5
Epoch	1	1	3	3
Vaild steps	125	125	125	125

Table 15: Hyperparameters for our method in fine-tuned setting